CHAPTER III
METHODOLOGY

Method and alternatives that have been selected for designing the controller system are explained in this chapter. The block diagram of this system generally shown in Figure 3.1 below:

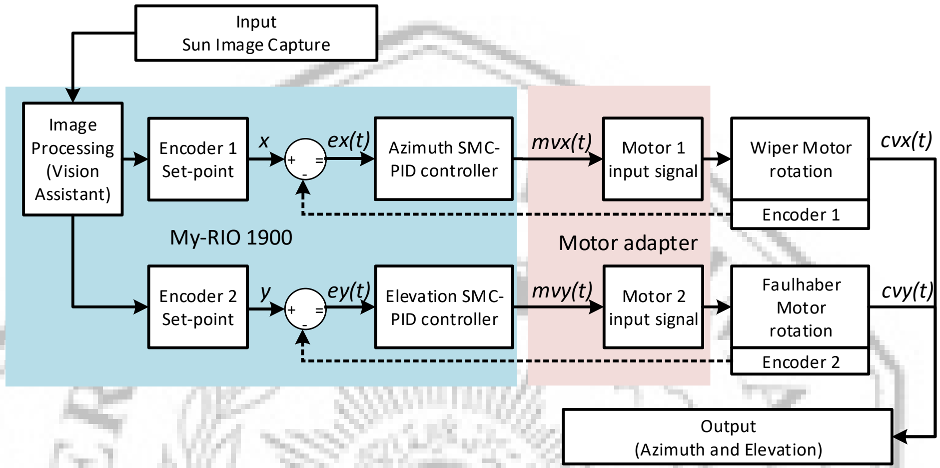

Figure 3.1 Block Diagram of Azimuth and Elevation Tracking System

The block diagram system in Figure 3.1 is a form of closed-loop control system. The initial input signal comes from the camera that captures the sun position image, then processed by MyRIO to find out the pixel values of the sun shift from the midpoint camera. The pixel values are converted into *x* and *y* in the encoder change value, which will be the real setpoint for the controller. The setpoint will be compared to the feedback from encoder by the summing block to get the error signal *e(t)*. The error signal is used as reference value for the control system (SMC-PID) to generate manipulated variable *mv(t)* signal in PWM value. PWM will be processed by motor driver to rotate the DC motor, so the controlled variable *cv(t)* that is azimuth and elevation will be rotated as well as the setpoint *x* and *y*.

The actuator consist of *wiper* motor to change azimuth position and *faulhaber* motor for elevation angle change. The wiper motor is used to change the rotation in *x* axes, while the faulhaber for *y* axis rotation. The clockwise or counterclockwise rotation is controlled by motor driver which gets intructions from control algorithm programmed into MyRIO.

3.1 Hardware Design

The design of hardware is arranged as Figure 3.2 below. There are several hardware used in this final project which can be known from this figure.

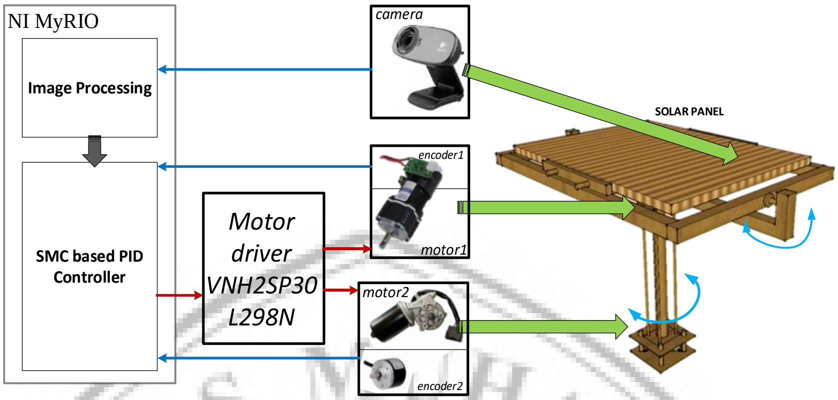

Figure 3.2 Hardware Design

The hardware design is supported with Figure 3.3 which shows the input-output pin used on the hardware. The wiring diagram also represented in this figure, the MXP & MSP connector used to connect the electrical module with MyRIO 1900. Pin connection of MyRIO to the all electrical devices is detailed in Table 3.1.

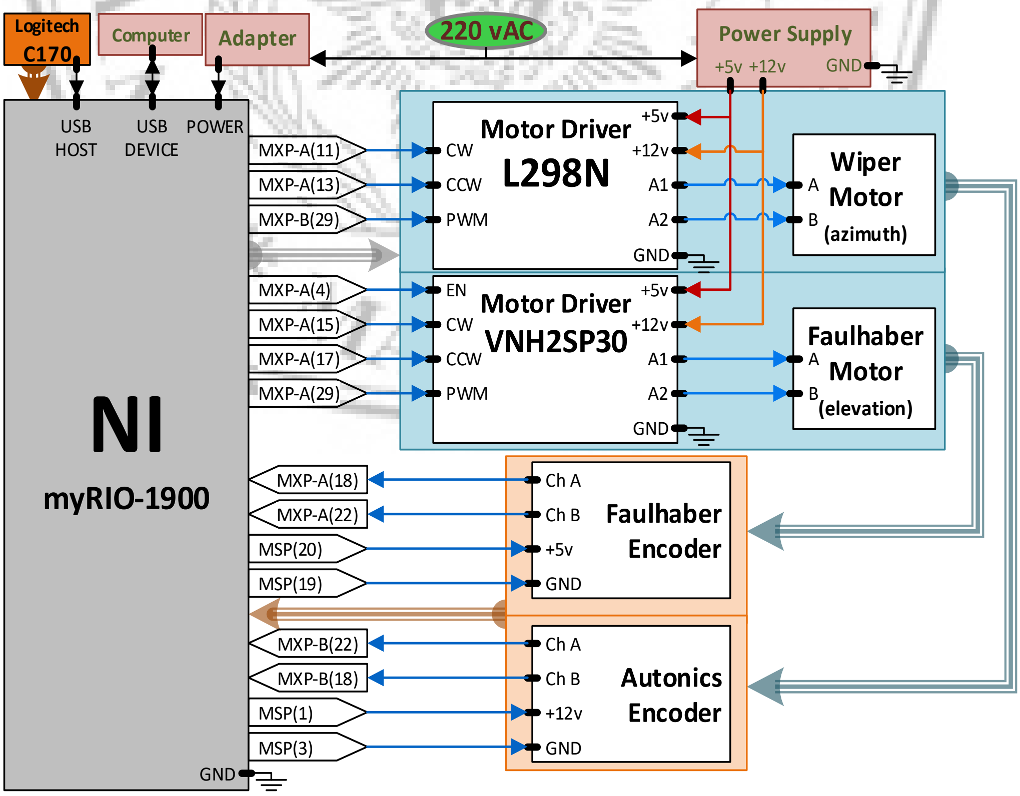

Figure 3.3 Wiring Diagram

Table 3.1 Input-Output Pins Connection

No.	MyRIO Pin Description		Connected to
	Type Conn.	Pin Number	
1	MSP	1	12 volt autonics encoder
2	MSP	3	Ground autonics encoder
3	MSP	19	Ground faulhaber encoder
4	MSP	20	5 volt faulhaber encoder
5	MXP-A	4	ENABLE VNH2SP30
6	MXP-A	11	CW L298N
7	MXP-A	13	CCW L298N
8	MXP-A	15	CW VNH2SP30
9	MXP-A	17	CCW VNH2SP30
10	MXP-A	18	Channel A faulhaber encoder
11	MXP-A	22	Channel B faulhaber encoder
12	MXP-A	29	PWM VNH2SP30
13	MXP-B	18	Channel B autonics encoder
14	MXP-B	22	Channel A autonics encoder
15	MXP-B	29	PWM L298N

3.2 Encoder to Angle Conversion

The value of encoder as feedback path is in integer value as the rotation position based on incremental encoder. To obtain the exact angular rotation value based on encoder value, calibration process is needed. That is the conversion from encoder value to degree magnitude, the selected method is by comparison (ratio).

3.2.1 Elevation Conversion

The LabVIEW graphical programming for elevation encoder conversion can be seen in Figure 3.4 below. The actual hardware value is in encoder value, then converted by calculation to find the elevation value in degree units value.

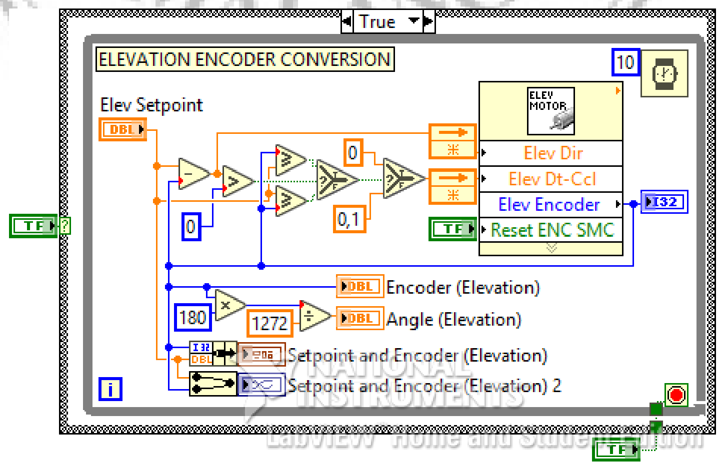

Figure 3.4 Graphical Programming for Elevation Encoder Conversion

3.2.2 Azimuth Conversion

Azimuth conversion has the same methode as elevation conversion. Figure 3.5 shows the LabVIEW graphical programming of azimuth encoder conversion. The true case program for elevation execution while the false case program used for azimuth execution.

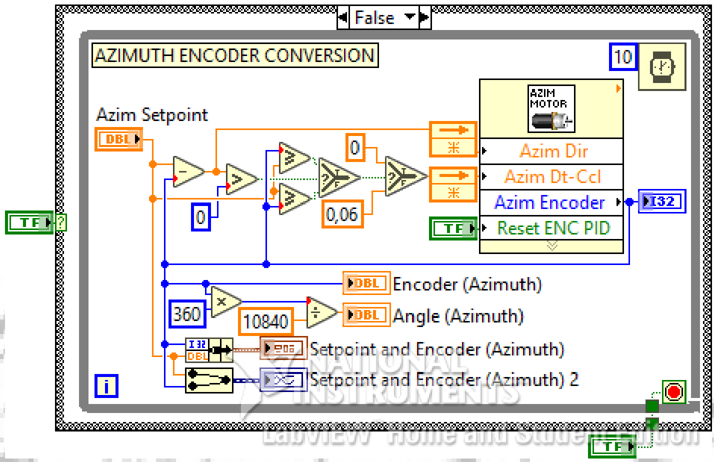

Figure 3.5 Graphical Programming for Azimuth Encoder Conversion

Figure 3.6 shows the LabVIEW front panel for elevation encoder conversion (at the left side) and azimuth (at the right side). The several gauges are used to facilitate the analysis of hardware conditions with encoder values detected. The maximal value of every gauge is adjusted to the elevation and azimuth ratio equation for the maximum angle.

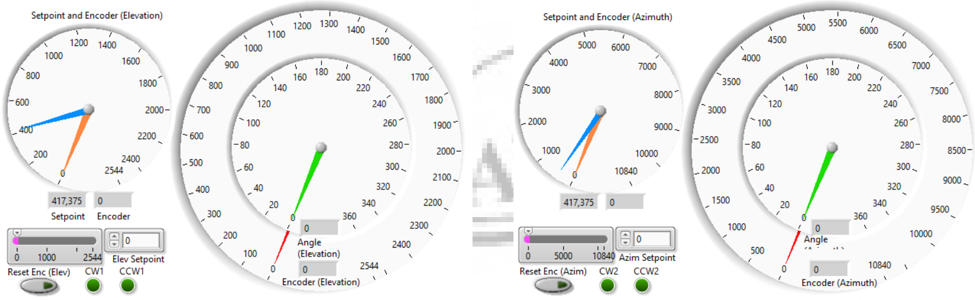

Figure 3.6 Front Panel for Encoder Conversion

3.3 Image Processing

Camera is used to take the sun picture that will be input for the image processing. The purpose of image processing is to determine the sun centroid location that will be represented in the cartesian coordinate of camera frame by the value of *x* and *y*.

3.3.1 Sun Detection

Sun position detection steps in this final project is consist of three steps, i.e. *Vision Acquisition* (capture the image with camera from MyRIO), *Vision Assistant* (process the image), and determination of sun shift position. The graphical programming to get the *x* and *y* value of the camera connected to MyRIO can be seen in Figure 3.7 below.

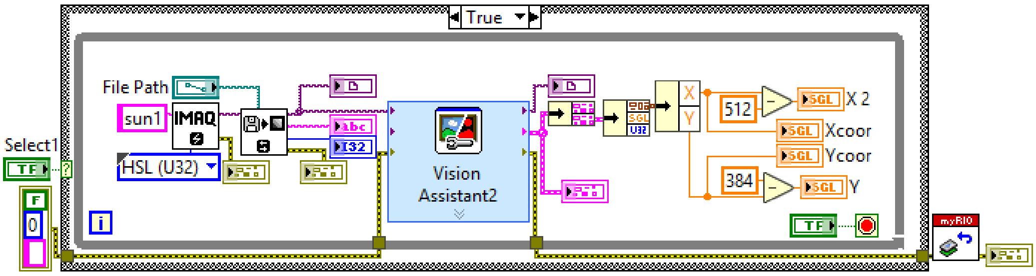

Figure 3.7 Labview Graphical Programming for Sun Position Detector

The detection process (from the original image until get the *x* and *y* coordinate values) is done by the *Vision Assistant*. The *Vision Assistant* program consist of 5 steps, i.e. thresholding, binary inversion, removing small objects, low pass filter, circle detection, and coordinate system determining. Figure 3.8 shows the program for sun position detection.

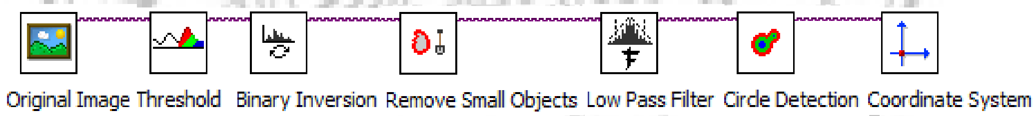

Figure 3.8 Block Diagram of Image Processing in NI Vision Assistant

The *Threshold* block uses *Color Threshold* feature for color image conversion and places the result into an 8-bit image. The captured image will be converted into greyscale to find higher contrast between the sun and other object. The selected grayscale pixel ranges is HSL based on *Hue*, *Saturation* & *Luminance*.

The *Binary Inversion* block is used to reverse the threshold result, it is used to reinforce the values of red pixel that means it's the sun position. To eliminate other small objects, the feature used is *Advanced Morphology*. Then *Low Pass Filter* to eliminate the frequency of sun objects which are less clear on the sun edge. The *Circle Detection* is used to get the main circle object which represents the sun. Then the *Coordinate System* will generate the *x* and *y* values.

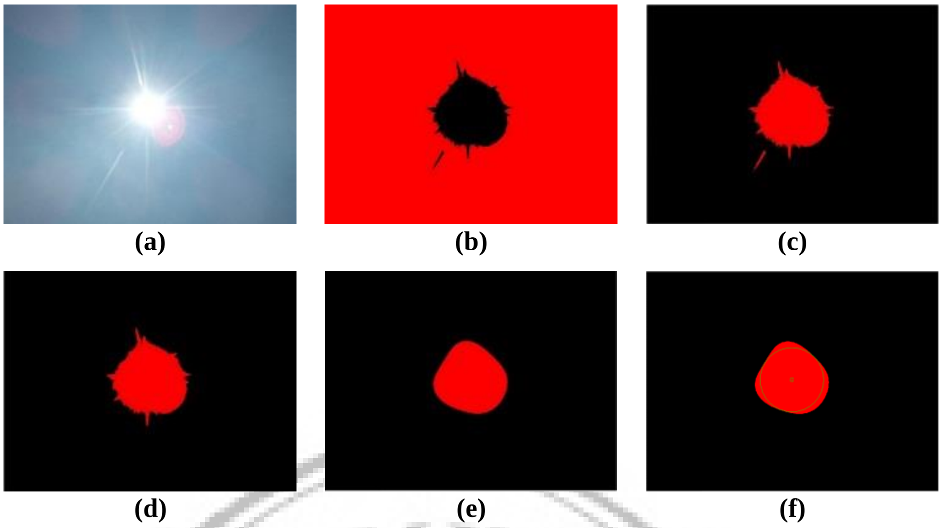

Figure 3.9 Sun Detection Process Image

There are several image in Figure 3.9 which shows the results of each image process sequence as Figure 3.8. Image (a) shows the original image of sun without cloud around it which will be processed by the *Vision Assistant*. Image (b) shows the threshold result which is consist of black and red color. Image (c) shows the binary inversion result which is the brightest area is the sun position display while the darker area is other objects.

Figure 3.9 (d) shows the advanced morphology result to remove small objects which removes the small objects that separate from the main circle. Image (e) shows the *low pass filter* result where the less obvious objects will be erased from the image. Then image (f) shows the *circle detection* result where we can see the position of sun circle. Then this circle processed by *coordinate system* to obtain the coordinate point from the center of the detected circle.

From the *Vision Assistant* result, the determination of sun shift position is shown in Figure 3.10 below. From this figure we can se the *x* error position from the center of camera is -2 and the *y* error position is 6.

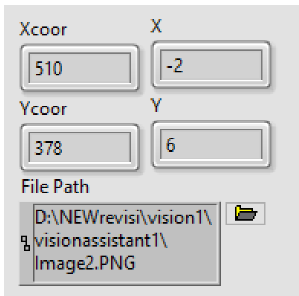

Figure 3.10 Sun Position Detection Result in *x* and *y* Value

3.3.2 Pixel to Encoder Conversion

The graphical programming to get the *x* and *y* value of the camera connected to MyRIO can be seen in Figure 3.11 below.

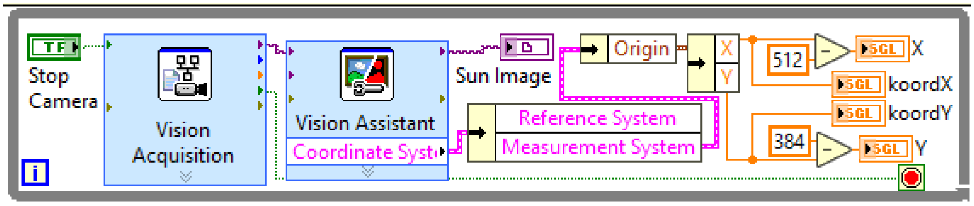

Figure 3.11 Graphical Programming for Sun Pixel Value

The original image obtained from *Vision Acquisition* block, then processed by *Vision Assistant* to get 2 axes coordinate system. The camera resolution is 1024x768, so the *x* result is reduced by 512 as the center point while 384 for *y* result.

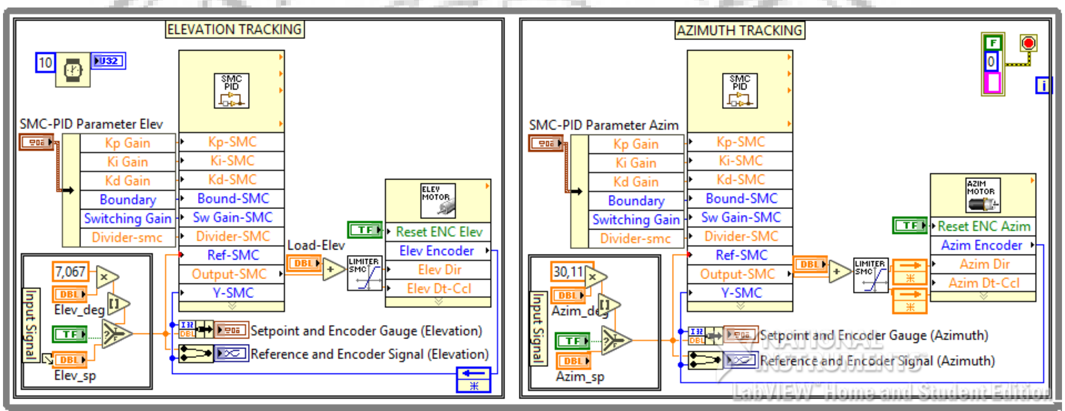

Figure 3.12 Graphical Programming for Pixel Conversion Tracking

Figure 3.12 shows the graphical programming to track the *x* and *y* error pixel from the center of camera frame.

3.4 DC Motor Modeling

The dentification of DC motor is a process to get the equation (mathematics model) of the actuator. Actuator modeling is required before designing the control system. SMC-PID using *State Space* structure model as dynamic equation, so from the modeling process must be obtained *State Space* equations as the initial step to design the control system. The general parameter of *State Space* is A, B, C, and D.

The identification process has several procedures, that is parameter estimation and validation. The parameter estimation aims to obtain the value of certain parameter from selected modeling structure by estimating the input-output relationship either into *Transfer Function* (TF) or *State Space* (SS) form. The identification procedures are performed with LabVIEW's existing features, i.e. *Identification System* in the *Control & Simulation* toolkit [17].

Modeling identification steps in this final project is consist of 4 steps, i.e. input-output testing, parameter estimation, validation, and response model testing. The first step is input-output testing with the varied setpoint value as input, while encoder response value as output that will be saved as *.tdms* file. Input-output data that has been stored, loaded by the *Identification Program* (Figure 3.13) to generate the *transfer function* model which is then converted to *state space*.

The second process is validation which is obtained by using mathematics feature in LabVIEW (that is Goodness of Fit.vi) which calculates 3 value i.e. *SSE*, *RMSE*, *R-square*, and *Best-fit*. The validation procedure is represented in percent (%) form called *best-fit*, it is the process of determining the accuracy level of DC motor modeling to the actual DC motor (hardware). The graphical programming of input-output testing for elevation can be seen in Figure 3.13, while for azimuth testing represented in Figure 3.14.

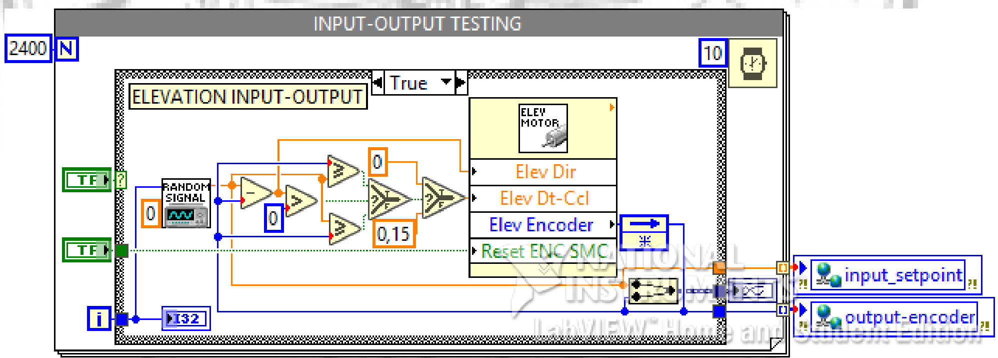

Figure 3.13 Graphical Programming for Elevation Input-Output Testing

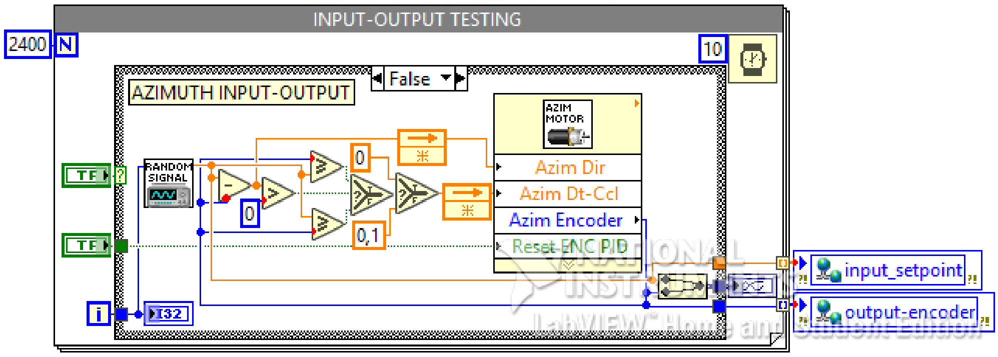

Figure 3.14 Graphical Programming for Azimuth Input-Output Testing

The hierarchy program of the Computer and MyRIO can be seen in Figure 3.15 (a). The shared variable in MyRIO can be read by Computer. While Figure 3.15 (b) shows the program to write data from MyRIO into Computer storage that will be saved in *.tdms* extension which can also be opened with Microsoft Excell.

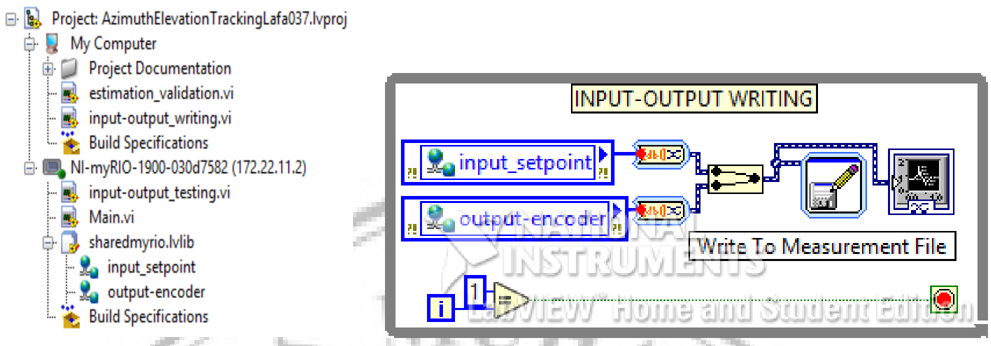

Figure 3.15 (a) Programming Hierarchy (b) Input-Output Writing

The estimation and validation modeling program shown in Figure 3.16 which generates TF and SS model. SS model is obtained from TF conversion result.

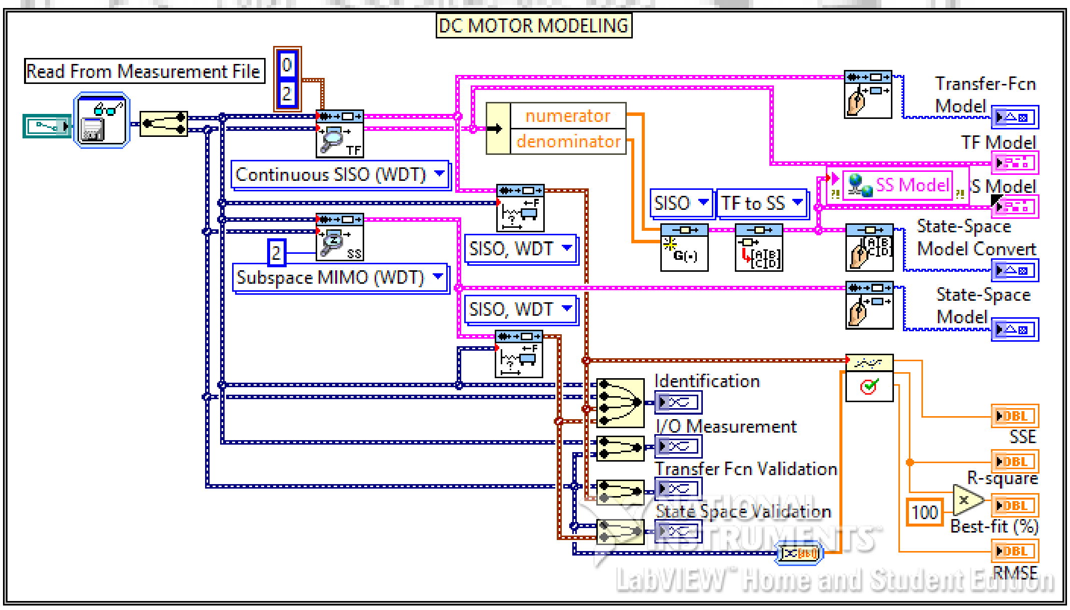

Figure 3.16 Graphical Programming for Estimation and Validation

Figure 3.16 shows the estimation block at the top while the validation block is at the bottom side. The important results taken are *State Space Model*, *Validating Curve SS*, and *Best Fit* value. All those results are obtained from the measurement file which is taken from input-output data saved of *Input-Output Writing Program* that is shown in Figure 3.15 (b) above.

3.5 SMC-PID

The implementation of SMC based PID sliding surface will be modeled as following block diagram in figure 3.17.

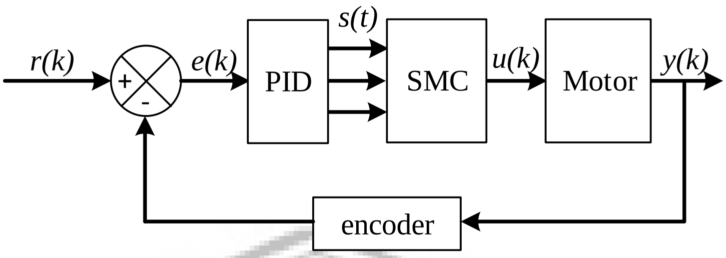

Figure 3.17 Block Diagram of Controller

The design is expected to reduce the effects of unwanted disturbances such as the mechanical load, wind and friction effects. After performing the DC motor modeling, the PID will be added to minimmize the rotational error between the motor output and the reference value. The PID parameter tuning is done by manual tuning in the LabVIEW *Control & Simulation* toolkit. Then SMC will be applied to the system where the PID will be integrated as the sliding function of SMC controller intended for chattering reduction. The interpretation of SMC is shown in Figure 3.18.

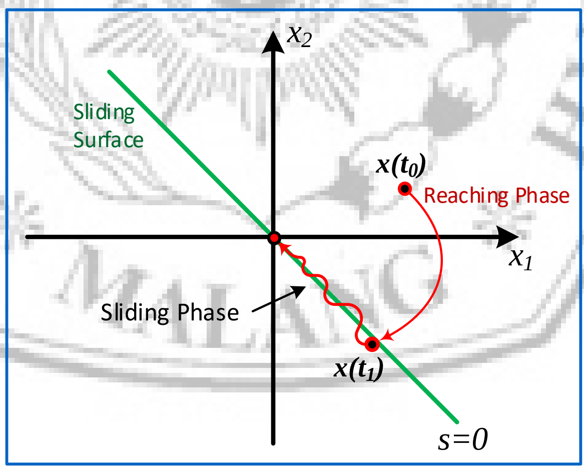

Figure 3.18 Sliding Mode Control Structure

The ***kp***, ***ki***, and ***kd*** values are the PID parameters as the determinant of the SMC sliding surface value. The mathematical equation is as follows:

$$s(t) = k_p e(t) + k_i \int_0^t e(\tau)\, d\tau + k_d \dot{e}(t) \tag{3.1}$$

The first derivative of the sliding surface equation above is expressed in equation (3.2).

$$\dot{s}(t) = k_p\dot{e}(\mathrm{t}) + k_i e(t) + k_d\ddot{e}(t) \tag{3.2}$$

The difference between the desired value with the actuator's actual position is formulated as the error value. The equation is expressed in the equation (3.3) below.

$$e(t) = x_r(t) - x_p(t) \tag{3.3}$$

The second derivative of the error equation above is expressed in the equation (3.4). This is coming from the second order model obtained through from the DC motor identification.

$$\ddot{e}(t) = \ddot{x_r}(t) - \ddot{x_p}(t) \tag{3.4}$$

The present value can be specified using the state space model parameters which is used. The second order of x_p is presented in equation (3.5).

$$\ddot{x}_p = -(A\dot{x} + Bx - Cu) \tag{3.5}$$

The *SMC control* (u_{SMC}) is consist of *switching control* (u_{sw}) and *equivalent control* (u_{eq}). The *switching control* is according to the reaching phase when $s(t) \neq 0$ while the *equivalent control* is according to the sliding phase when $s(t) = 0$. The general equation of SMC is denoted by equation (3.6).

$$u_{SMC}(t) = u_{sw}(t) + u_{eq}(t) \tag{3.6}$$

The general *switching control* formula based on *Lyapunov* theorem [17] is shown in equation (3.7). This theorem has been used to reduce the chattering effect which belongs to the usual u_{sw} formulas.

$$u_{sw}(t) = tanh\left(\frac{s}{\varphi}\right)k_s \tag{3.7}$$

The complete u_{sw} formula can be obtained by substituting equation (3.1) to equation (3.7). The result is expressed in the equation (3.8).

$$u_{sw}(t) = tanh\left(\frac{k_p \mathrm{e(t)} + k_i \int_0^t e(\tau)\, d\tau + k_d \dot{e}(t)}{\varphi}\right) k_s \tag{3.8}$$

Then the *equivalent control* can be obtained from the derivative of the *sliding surface* equation. This derivation equation is obtained by substituting equation (3.4) and (3.5) to equation (3.2). The formula is denoted in the equation (3.9) below.

$$\dot{s}(t) = k_p \dot{e}(\mathrm{t}) + k_i e(t) + k_d\{\ddot{x_r} + A\dot{x} + Bx - Cu\} \tag{3.9}$$

When $s = \dot{s}(t) = 0$, the *sliding control* will be executed because the sliding value has been reached. The equivalent control of SMC can be formed from equation (3.9) by assume the $\dot{s}(t) = 0$ that is represented in equation (3.10).

$$u_{eq} = \frac{1}{k_d C u}\left(k_p \dot{e}(\mathrm{t}) + k_i e(t) + k_d\{\ddot{x_r} + A\dot{x} + Bx\}\right) \tag{3.10}$$

3.5.1 Simulation Designing

The formulas can be applied to the LabVIEW *control & simulation* feature to ensure that the programming algorithms are properly constructed. Figure 3.19 show the Lab-VIEW simulation program for SMC-PID from the equations above. *KP*, *KI*, and *KD* in every block must have the same value, so the *local variable* feature used in this subsystem simulation program. In addition to SMC-PID, there is a classical PID controller programming block as a comparison of SMC-PID system robustness to interference.

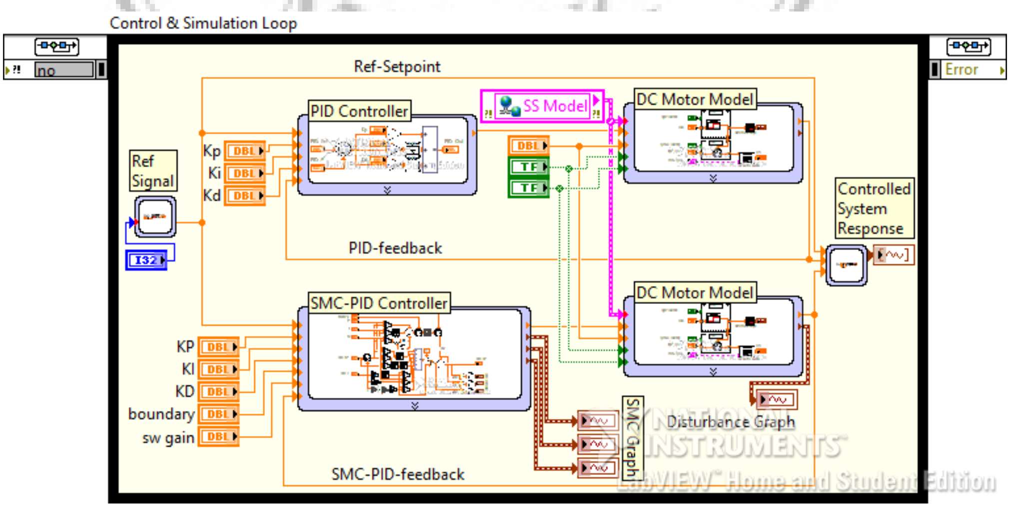

Figure 3.19 LabVIEW Programming for SMC-PID and PID Simulation

There are several *Simulation Subsystem* block in Figure 3.19 i.e. the *PID Controller*, *SMC-PID Controller*, and *DC Motor Model* which is shown in several picture below. Figure 3.20 shows the PID subsystem, Figure 3.21 shows the SMC-PID subsystem, and Figure 3.22 shows the DC motor model with disturbance block which is used to give the interference signal to the system.

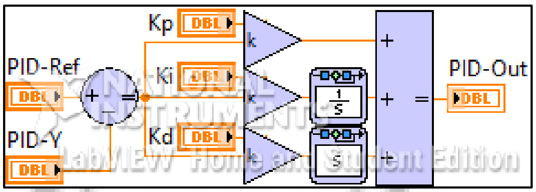

Figure 3.20 LabVIEW Simulation Subsystem for PID Controller

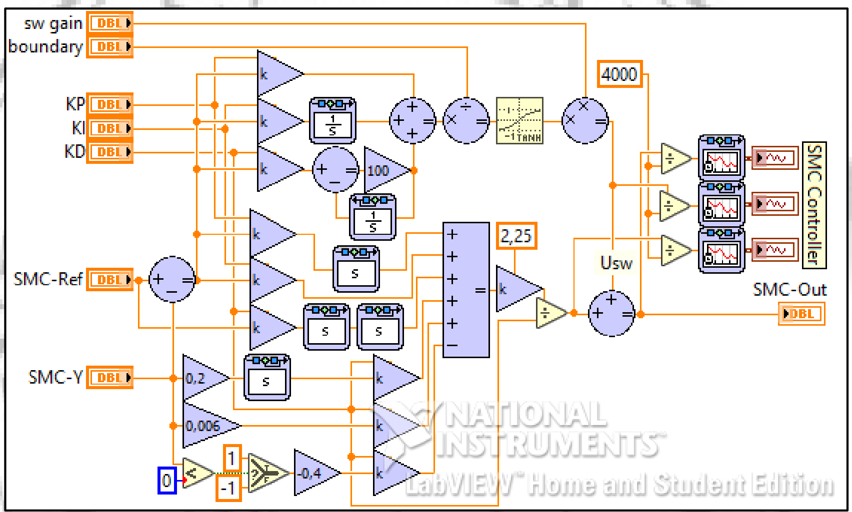

Figure 3.21 LabVIEW Simulation Subsystem for SMC-PID Controller

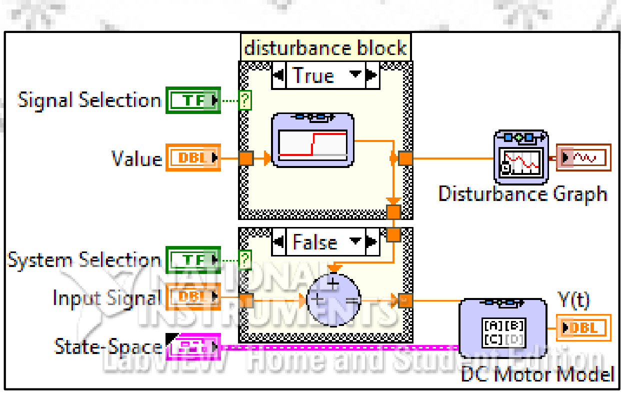

Figure 3.22 LabVIEW Simulation Subsystem for DC Motor Model

3.5.2 Hardware Validation

The hardware validation process is performed to ensure that the simulated control design can be actually applied. The elevation and azimuth validation processes are performed separately, which is tested using the PID and SMC-PID controller.

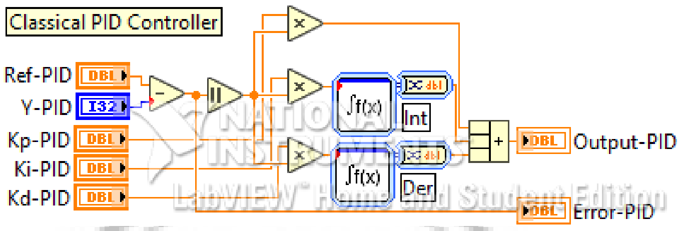

Figure 3.23 LabVIEW SubVi for Classical PID Controller

Figure 3.23 shows a manually arranged PID controller program without the PID feature in LabVIEW. While for SMC-PID can be seen in Figure 3.24 below.

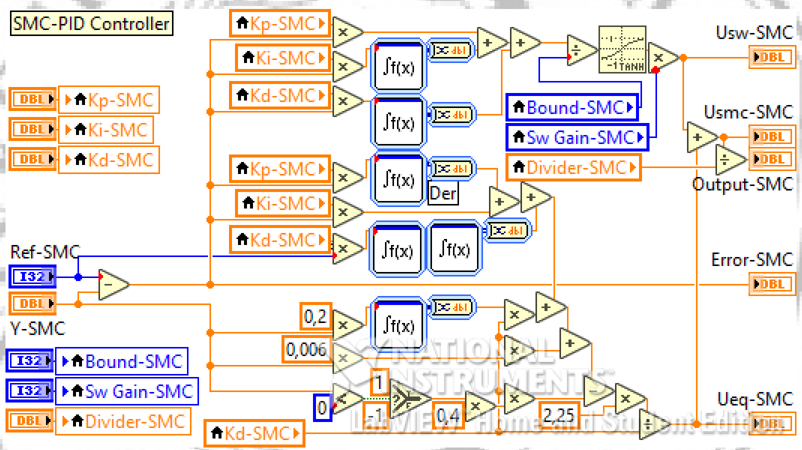

Figure 3.24 LabVIEW SubVi for SMC-PID Controller

3.5.2.1 Elevation

The elevation validation is performed to confirm the SMC-PID resistance compared to the PID against the artificial interference parameters and the actual load on the hardware. The disturbance parameter is a variable value which can be determined from the program. While the actual load is the natural weight difference of the elevation position at 0˚ (horizontal), 45˚, and 90˚ (vertical), which will affect to the DC motor torque. In horizontal position the torque is greater.

The PID graphical programming for elevation is can be seen in Figure 3.25 while the SMC-PID graphical programming in Figure 3.26.

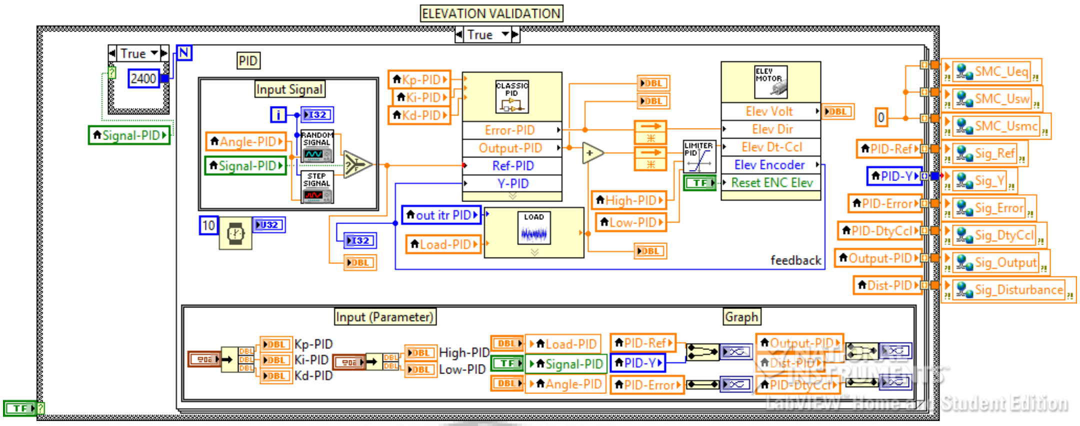

Figure 3.25 LabVIEW Graphical Programming for Elevation PID

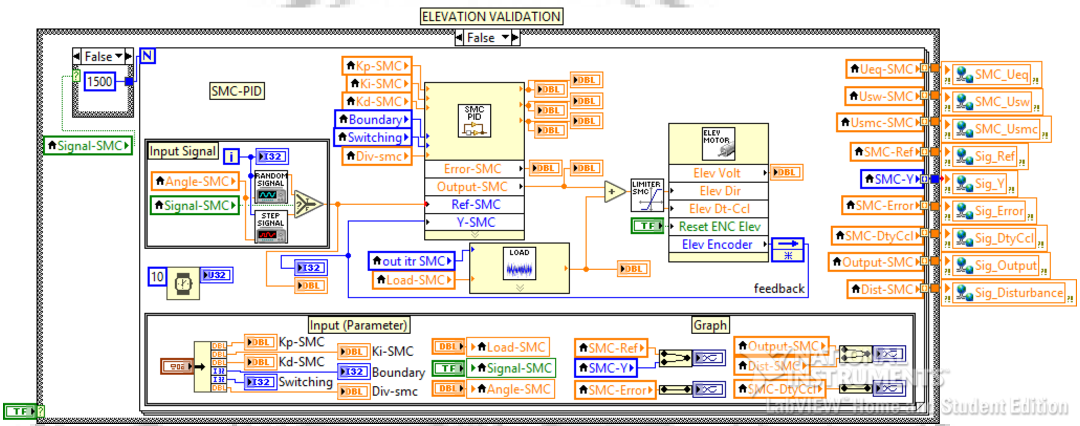

Figure 3.26 LabVIEW Graphical Programming for Elevation SMC-PID

The parameter settings and value for PID and SMC-PID for elevation validation is different. Every input parameters for the program execution is shown in Table 3.2 below.

Table 3.2 Parameters Setting Program for Elevation Validation

Elevation			
PID		SMC-PID	
Kp gain	750	*Kp gain*	4000
Ki gain	0	*Ki gain*	0
Kd gain	350	*Kd gain*	1000
Low output	0	*Divider*	5
High output	0,5	*Boundary*	15
Angle set	45˚	*Angle set*	45˚

3.5.2.2 Azimuth

The azimuth validation is performed to compare the SMC-PID accuracy with PID against the very sensitive sensor. In this process, the disturbance parameters are still given as done at the previous simulation process.

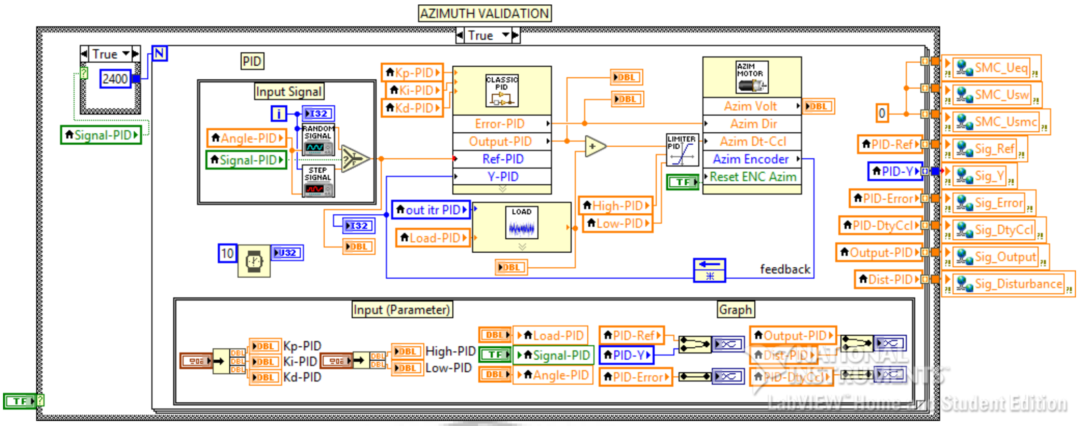

Figure 3.27 LabVIEW Graphical Programming for Azimuth PID

Figure 3.27 shows the PID graphical programming for azimuth, While the SMC-PID graphical programming is can be seen in Figure 3.28 below.

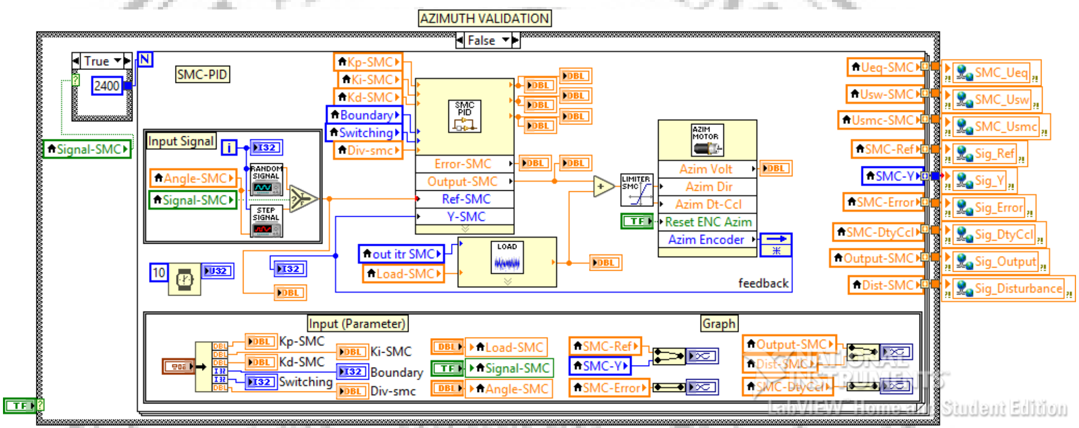

Figure 3.28 LabVIEW Graphical Programming for Azimuth SMC-PID

The parameter settings and value for PID and SMC-PID for azimuth validation is also different. Every input parameters for the program execution is shown in Table 3.3 below.

Table 3.3 Parameters Setting Program for Azimuth Validation

Azimuth			
PID		**SMC-PID**	
Kp gain	5000	*Kp gain*	60000
Ki gain	0	*Ki gain*	0
Kd gain	3700	*Kd gain*	20000
Low output	0	*Divider*	1
High output	0,1	*Boundary*	15
Angle set	0˚	*Angle set*	0˚

3.5.2.3 Validation Procedure

The validation procedure is the explanation of the steps and the way of hardware testing of the parameters provided. This process is divided into 3 kinds of testing i.e. *random signal input*, *step signal input* without the disturbance, and *step signal input* with the varied disturbance.

1) Random signal input

The amplitude and timing of the random signal are adjusted to the signal used during the *Input-Output Testing* during the *DC Motor Identification*. The random signal for azimuth can be seen in Figure 3.29. The given amplitudes are 60, 30, -30, 0, 45, and back to 0, those value will change sequentially every 4 seconds.

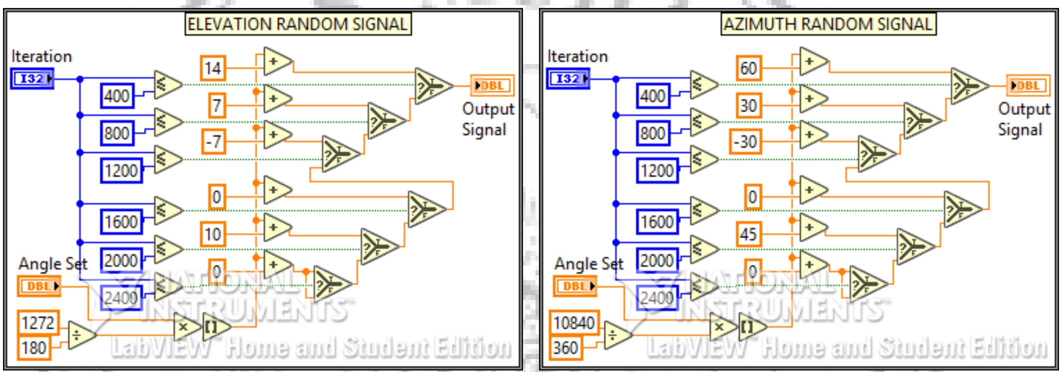

Figure 3.29 LabVIEW Graphical Programming SubVi for Random Signal

The amplitudes for elevation are 14, 7, -7, 0, 10, and 0, those values are the units for the encoder value. The starting angle of this test can be adjusted by changing the value of the *angle set* as determined in Table 3.2 and Table 3.3.

2) Step signal input

The amplitude and timing of step signal for azimuth can be seen in Figure 3.30 below. The amplitude given to the azimuth are 0, 30, and back to 0. The step signal starts from 0 to 30 after 1 second. It will be down to 0 after 11 seconds, but the data taken is only 10 seconds. While the elevation amplitude are 0, 7, and 0.

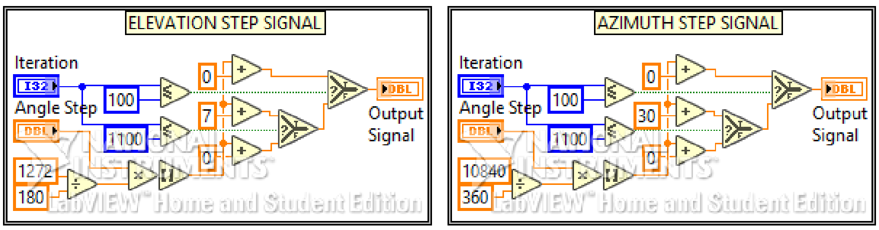

Figure 3.30 LabVIEW Graphical Programming SubVi for Step Signal

Table 3.4 Switching Gain Setting Program for Hardware Validation

SMC-PID *Switching Gain*			
Elevation		**Azimuth**	
Random	10000	Random	1500
No-disturbance	10000	No-disturbance	500
Disturbance 5000	30000	Disturbance 10000	11000

Table 3.4 indicates the *switching gain* value to be given during the test. The difference in *switching gain* values given aims to reduce the chattering which go on as long as the *switching function* works.

3.6 Overall LabVIEW System

The overall block diagram as the reference to design LabVIEW system can be seen in Figure 3.31. It shows how the overall software system works, which consist of input signal readings and it's calculation, control system, and controller output signal conditioning.

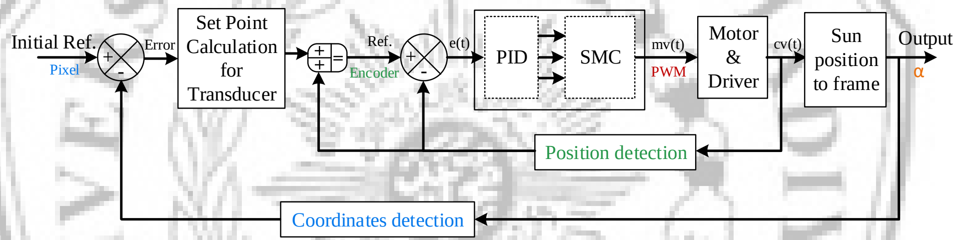

Figure 3.31 Block Diagram of Overall Control System

The LabVIEW graphical programming with SMC-PID control for both elevation and azimuth tracking system is represented in Figure 3.32 below. It consists of image processing loop with the encoder conversion block at the top side, while at the bottom side is the hardware tracking system.

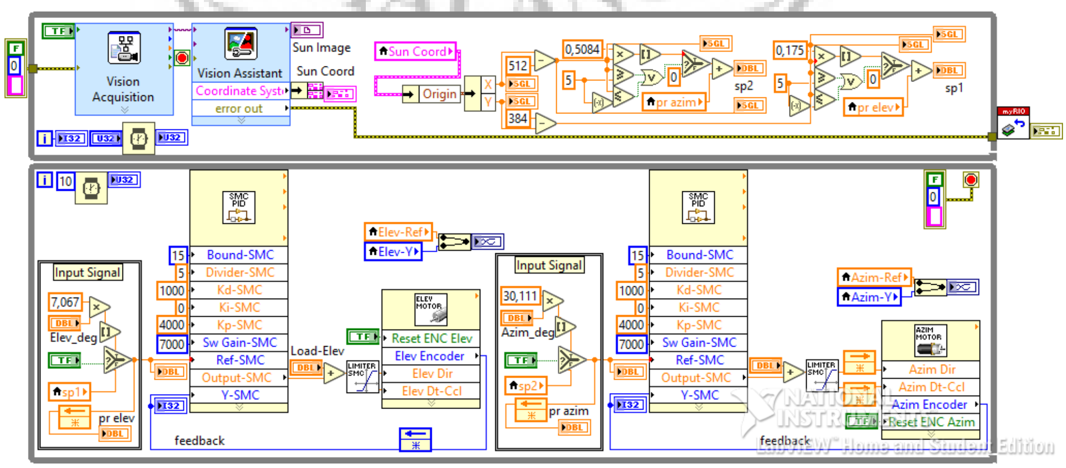

Figure 3.32 LabVIEW Graphical Programming of Overall Control System